

pragma's

product profiles

Issue #35

The *free* newspaper for Pick™ users.

May 28, 1987

Pragma's Product Profiles
is published periodically by
Semaphore Corporation,
207 Granada Drive,
Aptos, CA 95003,
(408) 688-9200.

Entire contents copyright © 1987
by Semaphore Corporation. All rights reserved. No
part of this publication may be reproduced in any form
or by any means without prior written permission from
Semaphore. Semaphore offers no warranty, either
express or implied, for any losses due to the use of any
material published.

Subscriptions are free, but are available only to Pick
users with USA mailing addresses.

All material received will be considered for publication.
Semaphore reserves the right to edit all submittals.

Semaphore cannot be responsible for lost issues or
issues returned to us because of address changes.

Pick is a trademark of Pick Systems.
Semaphore Corp. is not affiliated with Pick Systems.

Pragma's Product Profiles is completely FREE!

Name _____
Company _____
Address _____
City/State _____ ZIP _____
Telephone _____

I use the Pick operating system and my mailing address is in the USA.
Please continue sending me *Pragma's Product Profiles* free of charge.

Signature _____ Date _____

Return this form to:
Pragma • 207 Granada Drive • Aptos, CA 95003

Pragma's Product Profiles
207 Granada Drive
Aptos, CA 95003

BULK RATE
U.S. POSTAGE
PAID
APTOS, CA 95003
Permit No. 67

Address Correction Requested

We got a phone call the other day...

Telephone: Riiinnnnngggg!

Semaphore: Good morning, Semaphore Corporation.

Pat: Is Mike there, please?

Semaphore: One moment, please.

Mike: Hello, this is Mike.

Pat: Hi! This is Pat Snyder. Remember your old roommate?

Mike: Pat? Good grief! How you doing? Hasn't it been a couple of years since we last talked? Where are you now?

Pat: I'm fine. You're right, I think we last talked in '85. I've got a place in Pine Grove now. I'm kind of retired.

Mike: Retired? What are you talking about? Last I knew, you were doing that MRP project for the Army on a Microdata over at Fort Devens.

Pat: Heck, I finished that back in '84. Those Massachusetts winters were driving me crazy, so I moved back to San Francisco right after the project ended. I thought I sent you a postcard.

Mike: I don't think you did. Anyway, what's this retirement business?

Pat: You're not going to believe this. I hit six numbers on Lotto back in March! Actually, I matched five and the bonus number, you know, on the pick-six game? They gave me \$246,710.45! Well, split up in payments over twenty years...

Mike: Geez! That's amazing! I matched three numbers

once, but that only got me \$5.

Pat: No one was more amazed than me. I've been buying five \$1 tickets each week ever since Lotto started. I started out using a Zebra to pick the numbers.

Mike: You mean a General Automation Zebra?

Pat: Yeah. I was doing a one-day-a-week consulting job for a Western Union research office over by the Transamerica building up in the City, and they had a Zebra. Every Friday, I used a little BASIC program I wrote that called the RND function over and over until it picked six different numbers, and then it repeated the whole thing over again four more times so I could play five different tickets. I played new numbers every week. I wanted the machine to pick my numbers for me, mostly for psychological reasons. I assume California's number-picking machine is random. At least each week's numbers sure look random. You know, combinations come up that you would never think of picking yourself. But when I filled out a ticket by hand, I'd always steer towards certain numbers and avoid others, and what I'd end up with would always seem very non-random. My numbers would be all even, or bunched up at one end of the ticket, or something like that. So I figured I'd let the Zebra pick them for me. It sounds kind of silly now.

Mike: And the Zebra won you a quarter million dollars?

Pat: Not right off! In fact, one week I generated a set of tickets with the program, and the numbers looked familiar, like maybe I had played some of them before. Sure enough, I checked my old tickets, and all the numbers exactly matched a set of tickets I had generated two weeks previous! The darn RND function had generated the exact same sequence for me. I'd always known RND was pseudo-random, like all computer number generators, but it was too easy to force a repeat sequence. It must use the time of day as the starting seed, or some other stupid value that shows up too often. So I wrote my own random number generator to use instead.

Mike: And that's what won you the money?

Does Your Mailing Label Say RENEW?

12399/35 **RENEW**
MARSHA PICKUSER
555 MAPLE ST
APTOS CA 95003

If so, your subscription is about to EXPIRE! To avoid missing your next free issue of *Product Profiles*, renew IMMEDIATELY by filling out and returning the front cover coupon for a free subscription! The last two digits of your mailing label number indicate the number of the last issue you'll receive.

Tired of waiting for your
PickTM computer to SORT or
SELECT your large data files?

Need to quickly find *any*
attribute? Want to scroll files
up or down, in any sort order?

Now you can use B-TREE-P to instantly search, sort, and scroll any data from any Pick file!

Now you can instantly look up customers by name, street, Zip code, or any other field — not just by customer number. Now you can immediately find inventory entries by quantity, cost, or description — not just by part number. Whatever files you use, now you can instantly locate and display your data *any way you want*, without having to wait for endless SORTs and SELECTs.



Immediately display any record in any file just by typing one or more starting characters that match *any* field in the record.

You can display not only a selected record, but also any previous and next records, using *any* sort order you specify.

You can jump to any record in a file at any time, then browse through the file by scrolling up or down, a record at a time, or a page at a time, in *any* sort order.

Ask us for a free copy of Product Profiles #24, describing how B-TREE-P was originally developed and put to work. As one of Semaphore's programmers says: "We often ask ourselves why we waited so long to create B-TREE-P. After using it for our own production work, we wonder how we ever got by before without it. A Pick computer without B-TREE-P is like a car without wheels".

B-TREE-P is a proven collection of BASIC subprograms for using B-trees on Pick computer systems. B-trees allow any of the data in any of your Pick files to be instantly located and displayed in any sort order, without having to wait for SORT or SELECT commands.

B-TREE-P and a few minor modifications to your existing data entry programs are all that is necessary for you to immediately be able to search, display, and browse through your data quickly and conveniently.

Modifications to your existing data
programs are absolutely unnecessary!



Pick is a trademark of Pick Systems.

B-TREE-P includes all necessary
BASIC source code for a B-tree
system that works with any file:

- Insertion subroutine
- Deletion subroutine
- Lookup subroutine
- Previous/next subroutine
- Complete instructions

Plus, you receive the source code
for a complete demonstration
system that uses B-TREE-P to
maintain a name and address file:

Editor program for creating and
changing names and addresses.

Browser program for displaying
names and addresses.

Printer program for listing file
items in order without having to wait
for a sort.

Here's how to order:

Send your name, address,
telephone number, and your
check for \$395 payable to
Semaphore Corporation to:

Semaphore Corporation
207 Granada Drive
Aptos, CA 95003

We'll send you complete
B-TREE-P source code
listings and all necessary
documentation.

Call us at (408) 688-9200!

WARNING: B-TREE-P includes a license
agreement with copy, use, and transfer restrictions
limiting your use of B-TREE-P to one computer at a
time. Multi-CPU and OEM resale agreements are
also available.

Pat: Yup. I used the new generator for three weeks, and I hit the jackpot. It still seems like a dream.

Mike: What's the formula? Maybe it'll work for me...

Pat: [Laughs.] Well, I'll tell you how to code it, but don't get your hopes up. It's almost fourteen million to one against generating the right six numbers. Let's see. I used the formula Knuth gave us. Remember that numerical analysis class we took together? Choose parameters b , c , and m , then just pick some x as a seed, and use $(bx + c) \bmod m$ to generate a new x , which is your random number. Pump the new x back into the formula for the next x , and just repeat for as many numbers as you need.

Mike: What did you use for your first x ?

Pat: I used my birthday — the month, day, year digits — for my first x . My program just looped, generating x 's until I had thirty numbers for my five tickets. Then it saved the final x on disk in order to start the sequence again when I asked for another thirty numbers the following week.

Mike: The Lotto uses numbers from 1 to 49. Does that mean m is 50?

Pat: No, m should be a large power of 2, like maybe 262,144 or 1,048,576.

Mike: But then x ends up larger than 49.

Pat: Right. The most significant digits of x tend to be "more random", so right justify x and extract the first two digits on the left. If that number is from 1 to 49, use it on the ticket, else just generate the next x and try again.

Mike: Hmmm. If I use 2^n or 1,048,576 for m , then x ranges up to seven digits, so I right justify in a field of seven zeroes before extracting the two significant digits?

Pat: That sounds right.

Mike: What do I use for b and c ?

Pat: Pick c so it's odd, and not a multiple of five. Kind of like a recommended modulo for a Pick file! And c/m should also be approximately the square root of 3, divided by 6, subtracted from one half. That's approximately .2113248654051871, I think.

Mike: You've got to be kidding.

Pat: No, really! It's all in Knuth's book. And $(b \bmod 8)$ should be 5, b should be greater than the square root of m , greater than $m/100$, and less than m minus the square root of m . Also, avoid any pattern of digits in b 's binary or

decimal representation.

Mike: You mean, choose a string like the digits in an irrational number?

Pat: Exactly! Knuth mentioned using digits from π for b . I decided to use the log of π .

Mike: Amazing. Congratulations on hitting it big. What do you do now? Did you say you're in Pacific Grove?

Pat: No, Pine Grove, up in Amador county. It forms kind of an equilateral triangle with Stockton and Sacramento. Bought a real nice place. Lots of trees, clean air, no traffic. I'm sending you an invite for a housewarming party next month. I still do a little consulting, but I like to think of myself as retired. I found a good money manager, invested most of the winnings, and now I'm trying to live off the interest and take it easy. No more working for a living. Man, am I having fun.

Mike: I can't believe my ears. You used to live out of apartments and snack bar microwave ovens. I figured you would be the last person I know to settle down. What are you doing with all your new spare time?

Pat: Well, I've been doing a lot of swimming, getting a sun tan, doing a little traveling. I still do a lot of bike riding on my old Honda. About three nights a week, I've been toying with a little programming project I started. You know I've always liked to program...

Mike: Right. I think your motto was "stop me before I code again".

Pat: [Laughs.] Yeah, I've loved computers ever since I found out they existed. I was always amazed people were willing to pay me to program. Heck, I was having so much fun doing it, I was almost willing to pay them for the privilege! Anyway, I've been implementing the "perfect" database system on an IBM PC.

Mike: That's interesting. Tell me about it.

Pat: I call it "Tables", because it's very relational and everything is table-driven. A few important design concepts pervade the whole system. First of all, it's designed around a demand-paged virtual stack machine. I've designed a procedural, compiled language I call Twilight, because I always seem to be doing work on the system either at dusk or at dawn. The Twilight compiler is itself written in Twilight, and it generates pseudo-code interpreted by a multi-user kernel that's also written in Twilight. The idea is to keep the system highly transportable if it needs to be moved to another machine.

Mike: What kind of language is Twilight?



Experience an experienced computer system...and save!

If you are considering the purchase of an experienced computer system, call us. It can save you a pile of money.

We have a variety of systems available for quick delivery. We also stock disk drives, tape subsystems, memory communications controllers and anything else you might need.

We can provide you with useful information on manufacturers' product lines, policies and license fees. And, we will take your existing system in trade, but let you keep it during your conversion.

Microdata
Ultimate
ADDs

General Automation
Pertec
C. ITOH

Pick™ Systems
Fujitsu
and other Pick™-type systems

Also Printronix printers and others

CRTS — ADDs - Wyse - and others

Our technical staff is available to help you with any particular application.

New systems also available.

CARGILE & ASSOCIATES, INC.

783 Old Hickory Blvd., Suite 255A

Brentwood, TN 37027

(615) 373-2570

Where someone else's experience saves you money!

Pat: It's fairly close to Algol, because there's no variable typing like in Pascal. The only expression operators are add, subtract, multiply, divide, remainder, concatenation and substring extraction. For control I have block structure with scoped variables, IF/THEN/ELSE, and loops that use the form REPEAT X UNTIL (or) WHILE Y DO Z LOOP. That lets me do infinite loops with just REPEAT X LOOP. Unlike Pick, I like the word REPEAT to come first, kind of like a verb, and let LOOP come last, kind of like saying "loop back up to the beginning". I have very efficient, low-overhead module calls with parameters. Modules can be separately compiled and easily linked into the operating system's address space, so the system becomes very modular and extensible. The nicest thing about Twilight is that it has a very natural interface to the database. All record and field references are completely symbolic. In Tables, records are just strings in virtual memory that can be accessed by name or value at any time. There's no file system paradigm with OPEN's and file input/output and all that junk.

Mike: How about some examples?

Pat: Well, first let me describe how the database is structured. The data dictionary concept found in so many systems, including Pick, is a good idea, but it stops short. In Tables, I carry it to a logical conclusion in that every database "file" — that is, a table, a collection of records — has not just two parts (data and dictionary), but six distinct tables: data, fields, words, sorts, selects, and views. The data table simply contains arbitrary data records, like purchase orders or customer records. Naturally, records are a group of fields. One field is called the key field, and is guaranteed by the file creator to contain data unique to each record.

Mike: Like an item identifier in Pick?

Pat: Right, but with a lot less importance, since any kind of arbitrary lookup is fast and easy. The second part of every file is the fields table. It lists the symbolic name for each field in the data table. That lets records in the data table be accessed symbolically by Twilight code. For example, INVENTORY.QTY is a Twilight expression for the QTY field in an INVENTORY record. The compiler automatically checks the fields table to find the definition of the QTY field to know where to find that field in the INVENTORY record.

Mike: What about the equivalent of Pick attribute numbers?

Pat: There's no such thing in Tables. Only the system knows the physical structure and ordering of fields in a record. Programmers and users can only process records by symbolic names and expressions. In Pick, a list of attribute numbers defines fields. In Tables, the list of

symbols in the fields table defines the fields of a data table. Actually, it's the words table that contains the symbols and names normally used for accessing data fields.

Mike: Wait a minute, I'm getting confused. What's the difference between the fields table and the words table?

Pat: The fields table only names the available fields in the record, and indicates which one is the key field. For each entry in the fields table, there is one field in an actual data record. The words table is another table that lists all the words that can be used to get or put fields via Twilight input and output statements. The words table is very similar to the dictionary file in Pick: each symbol listed in the words table has a width, a justification, a label for columns on output and fields on input, an expression describing how to evaluate or convert the field, and so on. But there are a lot more parameters, and it's much more general. For example, instead of those horrible F and A-correlatives in Pick, expressions in the words table are compiled Twilight code that can even call whole programs if they want to. Even column labels are compiled expressions, instead of just constant strings.

Mike: What about the selects table?

Pat: The selects table is a list of different ways to select all or some of the records in the data table. It uses symbols from the words table to define selection criteria expressions, like the kind you usually see in a nonprocedural database language. For example, the selects table for a purchase order file might have one entry named HICOST, defined as ORDERCOST > "10,000". Another selection might be FIRSTQTR, defined as DATE >= "1/1" AND < "4/1". ORDERCOST and DATE would be symbols defined in the words table. Every selects table has at least one selection called DEFAULT, which is defined to include every record in the data table.

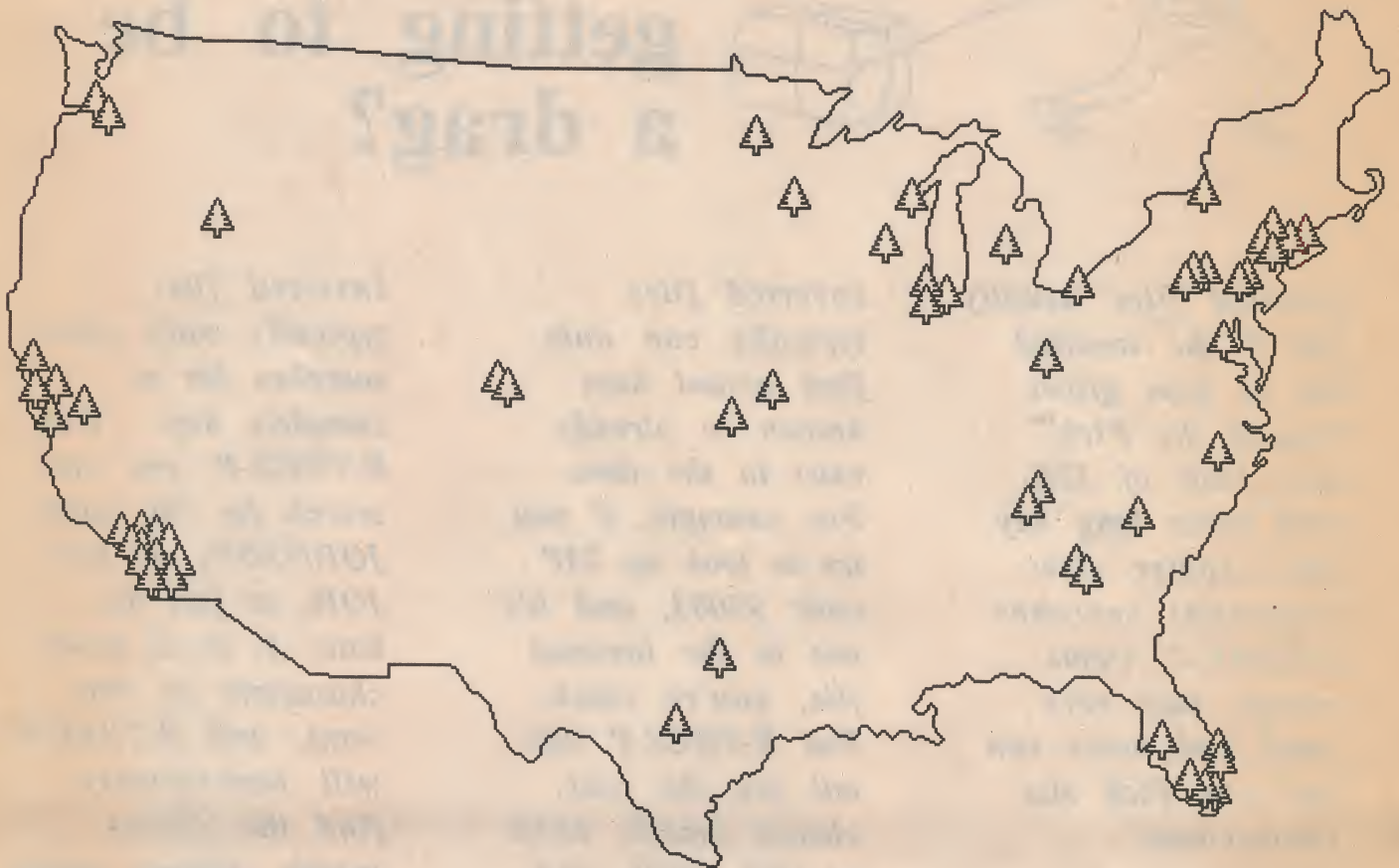
Mike: So I guess the sorts table lists the different ways to sort the data table?

Pat: Right, again using symbols from the words table. There might be a sorts table entry called BYCOST, defined as DESCENDING ORDERCOST. Another sorts entry might be BYVENDOR, defined as ASCENDING VENDOR ASCENDING ORDER. And every sorts table has at least one sort called DEFAULT, defined as ASCENDING <key>, where <key> is the actual symbol for the one guaranteed unique field in the data table.

Mike: I don't think I can guess what the views table does.

Pat: The views table lists all the ways to "view" the

New B-TREE-P installations are sprouting up every day!



Join the growing number of sites that are discovering the power and flexibility of B-TREE-P:

Long Beach Community Services • Halprin Supply Co. • Casualty Underwriters Inc. • University of California • Distributed Logic Corp. • Jet Electronics & Technology Inc. • NCAR • Trudell Trailer Sales Inc. • Stewart Co. • Computyme • Data Operating Systems Inc. • Tel-A-Train Inc. • Information Technology Consultants • Life & Health Insurance Co. of America • Flynt Systems Corp. • System Works Inc. • Miami Trading Enterprises • Generation Research • Multisystems Inc. • Infocel • Cooke Data Systems Inc. • John Klein & Assoc. Inc. • Condominium Insurance • Penn Independent Assoc. Inc. • Mark Card • Chandler Lumber Co. • Eye Care & Surgery Center • Chicago Kenworth • Wofford College • Conston Inc. • Assertive Systems • Office Works • Cornell University • City of Irvine • Excalibur Computer Systems Inc. • ADDS Inc. • Specs Music • Specialty Underwriters Inc. • Shoob Photography • May Trucking • Sierra Software Inc. • Medical Accounting Systems • AIPAC • NORPAC • Martin Cadillac Co. Inc. • Capital Software Ltd. • Oman Publishing • Reinsurance Assoc. • Hubert Distributing Co. • Livermore Police Dept. • Creative Computer Services • Berelson Co. • Minnesota Trade Office • XScribe Corp. • Glastron-Conroy Ltd. • Opportunities Unlimited • Topsy's International Inc. • ACS Systems Inc. • Bronson & Bratton Inc. • Access Software Inc.

*B-TREE-P is software that allows your data to be instantly located and output,
without having to wait for lengthy file sorts or searches.*

Write or call Semaphore Corporation, 207 Granada Drive, Aptos, CA 95003, (408) 688-9200.



Are your inverted files getting to be a drag?

Inverted files usually fail if the inverted list of keys grows beyond the Pick™ item limit of 32K. And those long key lists require slow sequential searches. B-TREE-P items always stay very small and never run into any Pick size limitations.

Inverted files typically can only find actual keys known to already exist in the data. For example, if you try to look up ZIP code 95003, and it's not in the inverted file, you're stuck. But B-TREE-P can tell you the next closest match, such as ZIP 95005, and even identify any size "neighborhood" of adjacent records before and after the match.

Inverted files typically only allow searches for a complete key. With B-TREE-P, you can search for the name JOHNSON, or just JOH, or just the letter J, or as many characters as you want, and B-TREE-P will immediately find the closest match, without using any disk space to duplicate the data.



Make your life easier! B-TREE-P can set you free today.



Inverted files typically cannot step sequentially through the index file. With B-TREE-P, you can start at any key, then step sequentially through all preceding or following keys.

Inverted files typically only allow searches for a primary key. With B-TREE-P, you not only can immediately retrieve the list of items all having, say, ZIP 10020, you can also immediately find the one item with ZIP 10020 and the address 207 MAPLE.

Software for inverted files is usually tied closely to the data being inverted, and must often be modified if a new type of data file or field is to be indexed. The "building block" routines in B-TREE-P never have to be modified, regardless of the application or type of indexing.

B-TREE-P is a product of Semaphore Corporation, 207 Granada Drive, Aptos, CA 95003, (408) 688-9200.
Pick is a trademark of Pick Systems.



data table during input or output. Each entry in the **views** table has a name, and specifies what selection criteria to use, what sort criteria, and what words (fields) to get or put in the **data** file. For example, the view called QTR1PARTS might specify the QTR1 selection, the BYPART sort, and the word list PART VENDOR PO. Then the Twilight statement PUT ORDERS QTR1PARTS will output (PUT) the purchase orders file (ORDERS) using the QTR1PARTS view. That view says to include only records in QTR1, output them BYPART, and list the columns PART, VENDOR and PO. Every **views** table has at least one view named DEFAULT that uses a DEFAULT selection (everything), a DEFAULT sort (by ascending key), and a word list equal to every field in the **data** table.

Mike: What's the complete syntax of the PUT statement?

Pat: All Tables input/output is done with GET/PUT statements of the form <verb> <file> <part> <view>. The <verb> is either GET or PUT. With GET, a table-driven editor executes and lets the operator create, change or delete records. With PUT, a table-driven output generator executes and just lists records on the screen or printer. The <file> is simply the name of any table in the system. All files and descriptions are listed in a master table called TABLES. The <part> is either DATA, FIELDS, WORDS, SORTS, SELECTS, or VIEWS, and is optional. If <part> is missing in a GET/PUT statement, then DATA is assumed. Remember, every file part is just another table, so the <part> component lets you edit or list any file part, not just the **data** part, kind of like the DICT phrase in a Pick LIST statement. Finally, the <view> is the name of a view in the **views** table, and is also optional. If <view> is missing, DEFAULT is assumed.

Mike: So PUT ORDERS means output the **data** part of the orders file using a default view?

Pat: Right. PUT ORDERS BYVENDOR means list the **data** part of the orders file using the BYVENDOR view. PUT ORDERS FIELDS means list the **fields** table with a default view. GET ORDERS WORDS BYWIDTH means input the **words** table for the orders file using the BYWIDTH view.

Mike: I think I'm beginning to see how to use your system. Have you created any actual applications software to test it all out?

Pat: I'm using a little accounts payable package as the acid test. Creating an application in Tables is really just a matter of getting all your data defined. The easiest way is to first just do CREATE calls to create the necessary files. Each CREATE requires only a name to use for the key field, and nothing else. The file creation routine allocates the six file parts from virtual memory, and leaves their

definitions in the master TABLES file. The **fields** table lists only the key field, the **words** table is empty, and the **selects**, **sorts** and **views** tables have only the default entries.

Mike: Let's say you created the VENDORS file, and the key field is VENDORNUM. What do you do next?

Pat: Use GET VENDORS FIELDS to start the editor and create the names of all the other fields in the vendors file. Then you can use GET VENDORS WORDS to create and edit words that refer to those fields. You might want the word LONGNAME to mean the NAME field left justified in 50 spaces, while the word SHORTNAME means the NAME field truncated to only 10 characters. After creating all desired words, use GET VENDORS SELECTS and GET VENDORS SORTS to define any necessary selection and sort criteria. Finally, use GET VENDORS VIEWS to define the different combinations of **sorts**, **selects**, and **views** for doing input and output. After that, use GET VENDORS to create or change vendors and PUT VENDORS to list vendors, tacking on the optional view name as desired to avoid the default sort and selection. For example, PUT VENDORS BYNAME would use the BYNAME view to output vendors alphabetically.

Mike: PUT sounds like it probably generates fairly typical columnar database reports, but how does editing work during a GET?

Pat: Actually, when PUT goes to the screen, you can do quite a bit more than just display each successive page. Remember, each view uses a given sort and selection criteria. Those correspond to a saved key list that is automatically built by the system when the criteria are defined, and automatically updated as **data** table records are created, changed, or deleted. As a result, there is never a wait for the sort and selection to occur, and the operator can use commands during a PUT to redisplay the first page of the listing (that's done from the keyboard with a control-Q), to redisplay the previous page (control-W), to display the next page (just return), or to display the last page (control-E).

Mike: Those also must work during a GET.

Pat: GET is oriented to records and fields, not display pages. The editor lets you jump to the first, last, previous, and next records in the view. Within a record, you can jump to the first, last, previous and next fields. The editor makes no decisions about how many records or fields to display at a time on the screen, or in what format. That's all defined in **words** table definitions referred to by the view when the GET statement is given. One view might let GET edit only three fields of a small group of records, one record at a time on the screen. Another view might pack ten records on the screen, showing five fields for

Unlock The Secrets In Your Computer!

Pragma (not to be confused with *Pragma's Product Profiles*) is the original 48-page technical journal for Pick users published quarterly beginning in August 1982. Each issue is packed with software and helpful information, including complete and debugged program listings and detailed, explanatory articles for readers at all levels of experience. Order your **Pragma** issues today and begin unlocking the secrets in your Pick system!

Pragma #1: Welcome to Pragma * ZIP Code File Design * A Program for Renumbering Statement Labels * Deriving Year-to-Date Counts * VANILLA, The No-Frills Manufacturing System: The Parts File * Moletron User Profile * More Memory, Fewer Reads * SYSMAP: A Cross-Reference System * How to Flag Missing Checks * 23 Wish List Items * Computing Modules with ENGLISH * Building ENGLISH Headings with Proc * 4 Queries * Making Item Identifiers Hash Well * Tape Handling * Patching the Exit Problem in 3.2 SCREENPRO * The Shell Game.

Pragma #2: We Have Liftoff * A Module Setting Program * Black Box Formatting * TRIM, DELIM and PROFILE Utilities * SYSMAP, A Cross-Reference System: File Format Input * Galinski Hamburg User Profile * LOOP vs. LOCATE Benchmarks * 25 Wish List Items * An Introduction to ENGLISH: Jargon * Proc Conversions * VANILLA, The No-Frills Manufacturing System: Bill of Material Input * 14 Queries * The Trouble Tree * 2 Letters * A Switchbox for 32 Ports * Security and the DATA/BASIC Programmer * The Cookie Game * Some New Subscribers.

Pragma #3: Is Pragma a Rare Medium, Well Done? * Edit Aides * A Proc for Cross-Referencing Q-Pointers * SYSMAP, A Cross-Reference System: Remaining File Input * Rainbow Natural Foods User Profile * More BASIC Benchmark Comparisons * Justifying Ragged Output * 13 Wish List Items * Generating Monthly Column Headings * VANILLA, The No-Frills Manufacturing System: Purchasing * 5 Queries * An Introduction to ENGLISH: More Commands * Shared Site Checklist * Converting Paint to Programs * 3 Letters * Generating Blank Forms * Animal, A Game That Learns * More New Subscribers.

Pragma #4: Survey Says... * Pacific Valley Bank User Profile * SYSMAP, A Cross-Reference System: Dicts and Procs * Uncompiling: Unassembling Stack Code * Left vs. Right Justification Benchmarks * 4 Wish List Items * A Comparison of BASIC Implementations * More New Subscribers * An Undocumented Editor Capability * 3 Local User Group Reports * Avoiding Saved Lists with PQ-RESELECT * Self-Documenting Reports * A Query * A Survey of Hardware that Supports Pick-Style Software * Printer Trade-Offs * An Introduction to ENGLISH: Finding Files * A Letter * VANILLA, The No-Frills Manufacturing System: Purchase Order Entry * The Swat Game.

Pragma #5: Happy Birthday! * Interactive Systems Producer Profile * Uncompiling: Regenerating Source Code * A Day of Revelation * IBM Personal Computer vs. Microdata Reality Benchmarks * VANILLA, The No-Frills Manufacturing System: Receiving * 3 Wish List Items * An Introduction to ENGLISH: Being Choosy * Converting Manual Paint to Programs * 6 Local User Group Reports * More New Subscribers * A Program That Reports File Pointer Locations * Boiler Plate Processing with Runoff * A New Query and an Old Query Answered * SYSMAP, A Cross-Reference System: Automation * Tape Types * 6 Letters * Permuted Index to the First 4 Issues of Pragma * Amazing, a Maze Game.

Pragma #6: Pick Pie Pictured * The Ubiquitous POINTER-FILE * Uncompiling: Resolving Labels * An Introduction to ENGLISH: Syntax Overview * AccuSoft Producer Profile * Rounding Out 7 Benchmarks On 5 Machines * Designing Data Entry Programs * 12 Wish List Items * GET: An Input Processor * 5 Local User Group Reports * More New Subscribers * Do You Know Your Proc Limits? * VANILLA, The No-Frills Manufacturing System: Inspection * 2 Queries * Individual Accounts or Shared Accounts? * Lock Logic Illustrated * Password Protection * Two Undocumented Conversions * A Letter * How AMAZING Works.

Pragma #7: More New Subscribers * A New Machine Visits Pragma * Bantam Hardware Overview * Suppressing LOCKED Clauses * An Introduction to ENGLISH: WITH, BY and TOTAL * Bantam Producer Profile * VANILLA, The No-Frills Manufacturing System: Disposition * A Bantam Diary * New Benchmark Timings for 8 More Machines * 6 Local User Group Reports * GET: Source Code * A Preprocessor for Symbolic Statement Labels * 2 Wish List Items * 3 Queries * Bantam Software Overview * A Letter * The Slide Game.

Send \$25 for each 48-page back issue to:

Pragma

207 Granada Dr. • Aptos, CA 95003



ONCE UPON A TIME, there lived a Pick™ computer user named Linda.

Every workday, from eight to five, Linda would sit in front of a terminal and type commands to make her computer produce reports. Sometimes the commands worked quickly and would make Linda's computer instantly display results. But usually Linda had to type a command that began with the dreaded words SORT or SELECT. Then the computer would take forever to process the command, and Linda would have to wait a long time before the computer could display the report.

Fortunately, Linda could stay busy while waiting for a SORT, because that's when she would always get lots of phone calls from her coworkers, who wanted to know why the displays on their terminals were suddenly slowing down to a crawl.

As each day passed, Linda got more and more bored with her slow computer.



One day, something terrible happened. Linda had just finished waiting ninety minutes for a complicated SORT, and was paging through the report on her terminal.

Suddenly, Linda accidentally hit the Return key, and page four of the report flashed by before she could read it.

Linda would have to do the whole SORT over again just to see page four.

Linda almost had a nervous breakdown.

Fortunately, Linda pulled herself together. But Linda was mad. She just wasn't going to put up with those slow SORTs and SELECTs anymore.

So Linda bought B-TREE-P.

Now Linda is very happy. She doesn't have to wait for her computer to SORT. Linda can instantly find and display any data she's looking for. Even scroll forward and backward through her files.

And the computer isn't sluggish anymore.

The moral of this story? Buy B-TREE-P. You'll be as happy as Linda.

each record, and allow edits on all fields. One of the important ideas in Tables is to allow editing of an entire database in the same easy way the operator might use a spreadsheet, but with complete customization allowed for the screen and printer formats.

Mike: If everything is so table-driven, why is the Twilight language necessary for applications programming?

Pat: There are many instances where logic has to be applied before or after some operator action. For example, creating a stock transfer record with the editor might cause a general ledger record to also be created. That kind of action has to be done by Twilight statements. Fortunately, the editor can automatically execute programmer-specified Twilight code before and after the editing of any field and before and after editing any record, so it's easy to create a completely custom application with lots of specialized logic.

Mike: You said Tables was a multi-user system, right?

Pat: As long as the host hardware has a millisecond timer, the software is easily configured to support any number of serial ports. I normally have it configured for only four processes during development. That's three ports and the spooler.

Mike: You have a spooler, too?

Pat: Sure, though it only spools on disk for optional output to a serial port. I don't think I'll ever bother supporting tape or parallel port output. The nice thing about the spooler is that it keeps all its information in standard tables, so it's very easy to alter or output the state of the spooler with GET and PUT statements. You don't need a lot of custom spooler commands. Plus, custom Twilight code can easily manipulate the spooler via the tables.

Mike: Just how much has the Pick system influenced you while developing Tables?

Pat: I really can't think of anything in Tables that originated with Pick. I've looked at a lot of database systems to compare ideas and try to separate the good from the bad. Remembering my own experiences as a programmer — fighting system software to try and make it do what I really wanted — that's been valuable. There's also an excellent book by Per Brinch Hansen that helped tremendously while designing Twilight and writing the kernel and compiler. *Programming a Personal Computer* is a very misleading title, but it's a book every systems programmer should have.

Mike: How would you say Tables differs from and is better than other database systems?

Pat: The most important difference is that it is very easy to "get under the hood" of Tables. It's not a system cast in concrete. Everything is written in Twilight and mostly table-driven. If you don't like the way the editor appears, you can completely change it. If you don't like the way file indices are structured, you can redefine them. If you don't like the host hardware, just recompile the kernel for a different target machine and move it over. If you think the system needs to offer a new conversion function, just compile it and link it in. One high level language gives you access to everything, and I mean everything. It takes me just a few minutes to do something like redefine the virtual memory frame size or maximum string length, and then completely recompile and reload the system.

Mike: Do you really think a programmer would find Tables that easy to modify?

Pat: I think most programmers would want to add to Tables, not so much change what's already there. One important feature that I really haven't mentioned so far is that everything in Tables is extensively and automatically cross-referenced. The editor and compiler automatically keep track of where every file and field is referenced. If you try to delete a field from a file definition, the system knows all programs, words, selects, sorts and views that refer to that field, and forces you to adjust and recompile them as necessary before the field can be deleted. It's essentially impossible for a programmer to monkey around with any code and leave behind a logically inconsistent system. That has a tremendous psychological effect that encourages a programmer to get into the system and try to change it for the better, without having to worry that bugs will creep in.

Mike: Are you going to sell Tables?

Pat: No. Tables is just a hobby for me, although I think a version I had earlier this year would easily qualify as a commercial product. I've toyed with the idea of maybe putting it on a public domain bulletin board somewhere, but I haven't bothered to do anything about it.

Mike: Pat, this call has been amazing. I'm already a half hour late for a meeting, so I'm going to have to run. I'll look for the invitation, and I'll be eager to see you again. Maybe by then your formula will have won a little cash for me, too.

Pat: [Laughs.] Talking's been fun. I'll see you at the party next month. Bye.

Mike: Bye.

Both telephones: Click.

The story you have just read is true. Or maybe it isn't. In any case, some names have been changed to confuse the situation even more.

FREE Back Issues

A few back issues of *Pragma's Product Profiles* are still available while supplies last. To receive your FREE copies, indicate which issues you need and send a stamped, self-addressed envelope to:

Pragma • 207 Granada Dr. • Aptos CA 95003

- Issue #10: Beyond The Power Of Pick
- Issue #11: Open Architecture Status Report
- Issue #12: Impressions Of The Zebra 750
- Issue #19: Pick Book A Disappointment
- Issue #21: Programming For Speed
- Issue #22: How Files Grow
- Issue #24: Beyond Pick, Revisited
- Issue #26: How To Find Wasted Space
- Issue #27: The Myth Of Separation=1
- Issue #28: Is BASIC Faster Than Access?
- Issue #29: How To Crunch Code
- Issue #30: Semaphore Interviews Itself
- Issue #32: This Month's Mailbag
- Issue #34: Sorting By Every Word In A Field

Now available by popular demand!

Our Pick™ Mailing List CREAM of the CROP



Our 3,000 most responsive Pick users' names and addresses. Only \$150 plus tax for one-up adhesive labels, sorted, printed, and delivered. Send no payment without first obtaining our order form and one-time use agreement.

P/Mail by Marianne

1893 Nadina Street • Seaside CA 93955

WE SPECIALIZE IN USED

Microdata™ Systems & Upgrades

Reality

128 MB Reflex II
50 MB Reflex I
128 K Mos Memory
Intelligent 8-Ways
Tape Drives
Complete Systems

Sequel M9000

IMB Memory
ACLC
Disc Drives
Controllers
Complete Systems

**BUY
SELL
TRADE**

Call **CHUCK HAAS**
For Competitive Prices

1-800-634-USED

EASTCOMP II INC.

513-528-5400

IN OHIO



**Semaphore Corporation
welcomes its newest
B-TREE-P customers:**

**Pactel Paging
United Collection Bureau Inc.
Laub Group Inc.
Associated Students UCLA
Brooks Equipment Co. Inc.
Anthony Pools
Samuelson Assoc. Co.
Educators Mutual Insurance
Magic Chef Air Conditioning Co.
Crosstern Corp.
Spectradyne Inc.
Tampa Bay Mgmt. Services Inc.
Northwest Agricultural Co-op Assoc.
Century Publishing Co.
J & L Industrial Supply Co. Inc.**